Representation Learning for Clustering: A Statistical Framework

Hassan Ashtiani
School of Computer Science
University of Waterloo
mhzokaei@uwaterloo.ca

Shai Ben-David
School of Computer Science
University of Waterloo
shai@uwaterloo.ca

Overview

Motivation

- There is a huge number of applications for clustering

Motivation

- There is a huge number of applications for clustering
- Tons of algorithmic choices
  - Clustering algorithms
  - Similarity metrics
  - Preprocessing techniques
  - Many conflicting outcomes
- How should we select among them?

Motivation

- There is a huge number of applications for clustering
- Tons of algorithmic choices
  - Clustering algorithms
  - Similarity metrics
  - Preprocessing techniques
  - Many conflicting outcomes
- How should we select among them?
  - Domain Knowledge
- How can such knowledge be incorporated into the clustering?
  - Trial and error?
  - Intuitions?
  - A more principled way?

Overview

Our Approach

Communicating Domain Knowledge

1. Take a small random subset of the data
2. Have a domain expert cluster the subset
3. "Learn" a model consistent with that clustering
4. Cluster the rest of data based on the model

- Central Questions
  - Any guarantees for the outcome?
  - How large should the sample be?
  - How shall models for clustering be represented?

Idea

- Rather than searching for an algorithm, we fix the algorithm and search for a suitable notion of similarity metric
- The algorithm we chose as our fixed clustering tool is k-means.
- Is this flexible enough?

Idea

- Rather than searching for an algorithm, we fix the algorithm and search for a suitable notion of similarity metric
- The algorithm we chose as our fixed clustering tool is k-means.
- Is this flexible enough?
  - Varying the metric over instances yields any possible data partition (I.e., k-means enjoys the richness property).

Idea

- Rather than searching for an algorithm, we fix the algorithm and search for a suitable notion of similarity metric
- The algorithm we chose as our fixed clustering tool is k-means.
- Is this flexible enough?
  - Varying the metric over instances yields any possible data partition (I.e., k-means enjoys the richness property).
- How can we avoid overfitting?

Idea

- Rather than searching for an algorithm, we fix the algorithm and search for a suitable notion of similarity metric
- The algorithm we chose as our fixed clustering tool is k-means.
- Is this flexible enough?
  - Varying the metric over instances yields any possible data partition (I.e., k-means enjoys the richness property).
- How can we avoid overfitting?
  - Select the metric from a specific class of candidate metrics

Idea

- Rather than searching for an algorithm, we fix the algorithm and search for a suitable notion of similarity metric
- The algorithm we chose as our fixed clustering tool is k-means.
- Is this flexible enough?
  - Varying the metric over instances yields any possible data partition (I.e., k-means enjoys the richness property).
- How can we avoid overfitting?
  - Select the metric from a specific class of candidate metrics
- What if the optimal metric is not inside the class?

Idea

- Rather than searching for an algorithm, we fix the algorithm and search for a suitable notion of similarity metric
- The algorithm we chose as our fixed clustering tool is k-means.
- Is this flexible enough?
  - Varying the metric over instances yields any possible data partition (I.e., k-means enjoys the richness property).
- How can we avoid overfitting?
  - Select the metric from a specific class of candidate metrics
- What if the optimal metric is not inside the class?
  - We will establish an agnostic guarantee!

Idea II

Communicating Domain Knowledge - Revisited

1. Take a small random subset of the data
2. Have a domain expert cluster the subset
3. Let the algorithm select a metric (from a class of metrics)
4. Perform k-means clustering using the metric and cluster the rest of the data

- What kind of algorithm should we use?
- What kind of guarantee can we expect?
  - We will establish **PAC-type** guarantees.

Overview

Previous Work

- Semi-Supervised Clustering
  - Constrained clustering (*must*/*cannot* links)
  - Modify the clustering objective (Demiriz et al. (1999); Law et al. (2005); Basu et al. (2008))
  - Metric learning (Xing et al. (2002); Alipanahi et al. (2008))
  - Mostly ad hoc, with focus on computational aspects rather than statistical guarantees
- Property-based Clustering (Ackerman, Ben-David and Loker, 2010)
  - Appropriate for selecting the algorithm
  - Properties are not yet user-level

Overview

Definitions

- X: The domain
- $f : X \mapsto \mathbb{R}^d$
- Learning the mappings is equivalent to learning similarity metric/kernel
- C_X^f: The clustering of X induced by first mapping the data by f and then doing k-means clustering
- $\mathcal{F}$: A class of mappings from X to $\mathbb{R}^d$
- C^*: Optimal (unknown) k-clustering of X
- Algorithm $A(S, C_S^*)$ takes a sample $S \subset X$ and its clustering C_S^*, and outputs a mapping $f_A \in \mathcal{F}$

Definitions II

- f_A may not be optimal. How can we measure its "error"?
- The error is the $\Delta_X(C^*, C_X^{f_A})$ (the difference between C^* and the clustering induced by f_A)
- f_A is ϵ-optimal when $\Delta_X(C^*, C_X^{f_A}) \leq \epsilon$

Definitions II

- f_A may not be optimal. How can we measure its "error"?
- The error is the $\Delta_X(C^*, C_X^{f_A})$ (the difference between C^* and the clustering induced by f_A)
- f_A is ϵ-optimal when $\Delta_X(C^*, C_X^{f_A}) \leq \epsilon$
- Agnostic ϵ-optimality:

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f) + \epsilon$$

Definitions II

- f_A may not be optimal. How can we measure its "error"?
- The error is the $\Delta_X(C^*, C_X^{f_A})$ (the difference between C^* and the clustering induced by f_A)
- f_A is ϵ-optimal when $\Delta_X(C^*, C_X^{f_A}) \leq \epsilon$
- Agnostic ϵ-optimality:

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^{f}) + \epsilon$$

- A natural choice of distance between two k-clusterings:

$$\Delta_X(C^1, C^2) = \min_{\sigma \in \pi^k} \frac{1}{|X|} \sum_{i=1}^{k} |C_i^1 \Delta C_{\sigma(i)}^2|$$

Problem Formulation

PAC Supervised Representation Learning for K-Means (PAC-SRLK)

A is a PAC-SRLK learner for $\mathcal{F}$ with $m_{\mathcal{F}}$ samples if

For every X and C^*, if S is a randomly (uniformly) selected subset of X of size at least $m_{\mathcal{F}}(\epsilon, \delta)$, then with probability at least $1 - \delta$

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f) + \epsilon$$

Problem Formulation

PAC Supervised Representation Learning for K-Means (PAC-SRLK)

A is a PAC-SRLK learner for $\mathcal{F}$ with $m_{\mathcal{F}}$ samples if

For every X and C^*, if S is a randomly (uniformly) selected subset of X of size at least $m_{\mathcal{F}}(\epsilon, \delta)$, then with probability at least $1 - \delta$

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f) + \epsilon$$

- Can we bound the sample complexity, $m_{\mathcal{F}}(\epsilon, \delta)$?

Problem Formulation

PAC Supervised Representation Learning for K-Means (PAC-SRLK)

A is a PAC-SRLK learner for $\mathcal{F}$ with $m_{\mathcal{F}}$ samples if

For every X and C^*, if S is a randomly (uniformly) selected subset of X of size at least $m_{\mathcal{F}}(\epsilon, \delta)$, then with probability at least $1 - \delta$

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f) + \epsilon$$

- Can we bound the sample complexity, $m_{\mathcal{F}}(\epsilon, \delta)$?
- Intuitively, the richer $\mathcal{F}$, the more samples we need.

Overview

TERM Algorithm

- What kind of algorithm can be a PAC-SRLK learner?

Transductive Empirical Risk Minimization (TERM)

A TERM learner for $\mathcal{F}$ takes as input a sample $S \subset X$ and its clustering Y and outputs:

$$A^{TERM}(S, Y) = \arg\min_{f \in \mathcal{F}} \Delta_S(C_X^f\Big|_S, Y)$$

- It finds the mapping based on which if you cluster X, the empirical error will be minimized.

Overview

Sample Complexity of PAC-SRLK

Theorem

The sample complexity of representation learning for k-means clustering (PAC-SRLK) with respect to $\mathcal{F}$ is upper bounded by

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \mathcal{O}\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right)$$

where $\mathcal{O}$ hides logarithmic factors.

- Pseudo-dimension measures the capacity of $\mathcal{F}$

Sample Complexity of PAC-SRLK

Theorem

The sample complexity of representation learning for k-means clustering (PAC-SRLK) with respect to $\mathcal{F}$ is upper bounded by

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \mathcal{O}\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right)$$

where $\mathcal{O}$ hides logarithmic factors.

- Pseudo-dimension measures the capacity of $\mathcal{F}$

Corollary

Let $\mathcal{F}$ be a set of *linear* mappings from $\mathbb{R}^{d_1}$ to $\mathbb{R}^{d_2}$. Then

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \mathcal{O}\left(\frac{k + d_1 d_2 + \log(\frac{1}{\delta})}{\epsilon^2}\right)$$

Overview

Sketch of the Proof

1. Bound $Pdim(\mathcal{F})$
2. Bound $\mathcal{N}(\mathcal{F}, d_{L_1}^X, \epsilon)$ based on $Pdim(\mathcal{F})$ and ϵ
3. Bound $\mathcal{N}(\mathcal{F}, \Delta_X, \epsilon)$ based on $\mathcal{N}(\mathcal{F}, d_{L_1}^X, \epsilon)$
4. Bound the $m_{UC}^{\mathcal{F}}(\epsilon, \delta)$ based on δ and $\mathcal{N}(\mathcal{F}, \Delta_X, \epsilon)$
5. Bound $m^{\mathcal{F}}(\epsilon, \delta)$ based on $m_{UC}^{\mathcal{F}}(\epsilon, \delta)$

Overview

Conclusions

- We proposed a framework for exploiting domain knowledge into clustering.
- We defined the notion of PAC-SRLK for the framework.
- The sample complexity of learning was bounded based on the pseudo-dimension of the class of mappings.
- The algorithm used to prove the result was a variant of empirical risk minimization.
- Open Problems
  - Computational complexity?
  - Generalizing the results to other clustering algorithms

Thank You!